Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 5

По дисциплине «Основы алгоритмизации и программирования»

На тему «Объединения, перечисления, битовые поля»

 Выполнила:

Студентка1 курса 6 группы

Альшевская Алина Михайловна

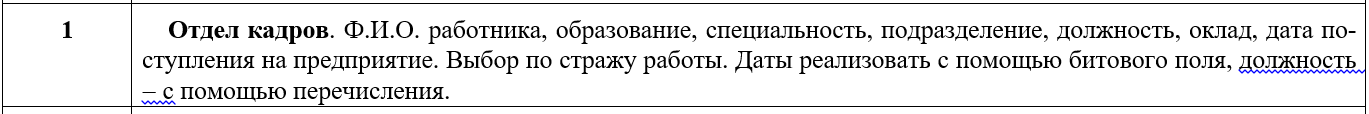
Преподаватель: асс. Андронова М.В.

2023, Минск

Вариант 1

Задание 1

В соответствии со своим вариантом разработать программу с использованием ***перечислений*** и ***битовых*** *полей* для работы с данными из таблицы, приведенной ниже. Реализовать функции ввода с клавиатуры, вывода на экран, удаления, поиска элементов. Интерфейс пользователя осуществить в виде меню.



#include <iostream>

#include <string>

using namespace std;

// Перечисление для должностей

enum Position {

MANAGER,

ENGINEER,

CLERK,

TECHNICIAN

};

// Структура для работника

struct Employee {

std::string names;

std::string education;

std::string specialty;

std::string department;

Position position;

double salary;

int day ;

int month ;

int year ;

};

Employee employees[100]; // Массив работников

int numEmployees = 4; // Количество работников

// Функция для ввода данных работника

void inputEmployee(Employee& employee) {

std::cout << "Введите Ф.И.О.: ";

cin.ignore();

std::getline(std::cin, employee.names);

std::cout << "Введите образование: ";

std::getline(std::cin, employee.education);

std::cout << "Введите специальность: ";

std::getline(std::cin, employee.specialty);

std::cout << "Введите подразделение: ";

std::getline(std::cin, employee.department);

std::cout << "Введите должность (0 - Менеджер, 1 - Инженер, 2 - Техник, 3 - Клерк): ";

int position;

std::cin >> position;

employee.position = static\_cast<Position>(position);

std::cout << "Введите оклад: ";

std::cin >> employee.salary;

std::cout << "Введите дату поступления на работу (день, месяц, год): ";

std::cin >> employee.day >> employee.month >> employee.year;

}

// Функция для вывода данных работника

void outputEmployee(const Employee& employee) {

std::cout << "Ф.И.О.: " << employee.names << std::endl;

std::cout << "Образование: " << employee.education << std::endl;

std::cout << "Специальность: " << employee.specialty << std::endl;

std::cout << "Подразделение: " << employee.department << std::endl;

std::cout << "Должность: ";

switch (employee.position) {

case MANAGER: std::cout << "Менеджер"; break;

case ENGINEER: std::cout << "Инженер"; break;

case TECHNICIAN: std::cout << "Техник"; break;

case CLERK: std::cout << "Клерк"; break;

}

std::cout << std::endl;

std::cout << "Оклад: " << employee.salary << std::endl;

std::cout << "Дата поступления на работу: " << employee.day << "." << employee.month << "." << employee.year << std::endl;

}

// Функция для удаления работника

void deleteEmployee() {

int del;

cout << "\nНомер работника, которого надо удалить " << endl;

cin >> del;

for ( del = (del - 1); del < numEmployees- 1; del++) {

employees[del] = employees[del + 1];

}

numEmployees = numEmployees- 1;

}

// Функция для поиска работника

Employee\* searchEmployee(Employee\* employees, int numEmployees, const std::string& name) {

for (int i = 0; i < numEmployees; ++i) {

if (employees[i].names == name) {

return &employees[i];

}

}

return nullptr;

}

int main() {

setlocale(LC\_ALL, "Russian");

while (true) {

std::cout << "1. Добавить работника\n";

std::cout << "2. Вывести информацию о работнике\n";

std::cout << "3. Удалить работника\n";

std::cout << "4. Найти работника\n";

std::cout << "5. Выход\n";

std::cout << "Выберите действие: ";

int choice;

std::cin >> choice;

Employee\* employee;

std::string name;

switch (choice) {

case 1: {

if (numEmployees < 100) {

inputEmployee(employees[numEmployees]);

numEmployees++;

}

else {

std::cout << "Нет места для нового работника\n";

}

}

break;

case 2:

std::cout << "Введите имя работника: ";

std::cin >> name;

employee = searchEmployee(employees, numEmployees, name);

if (employee != nullptr) {

outputEmployee(\*employee);

}

else {

std::cout << "Работник не найден\n";

}

break;

case 3:

deleteEmployee();

break;

case 4:

std::cout << "Введите имя работника: ";

std::cin >> name;

employee = searchEmployee(employees, numEmployees, name);

if (employee != nullptr) {

std::cout << "Работник найден\n";

}

else {

std::cout << "Работник не найден\n";

}

break;

case 5:

return 0;

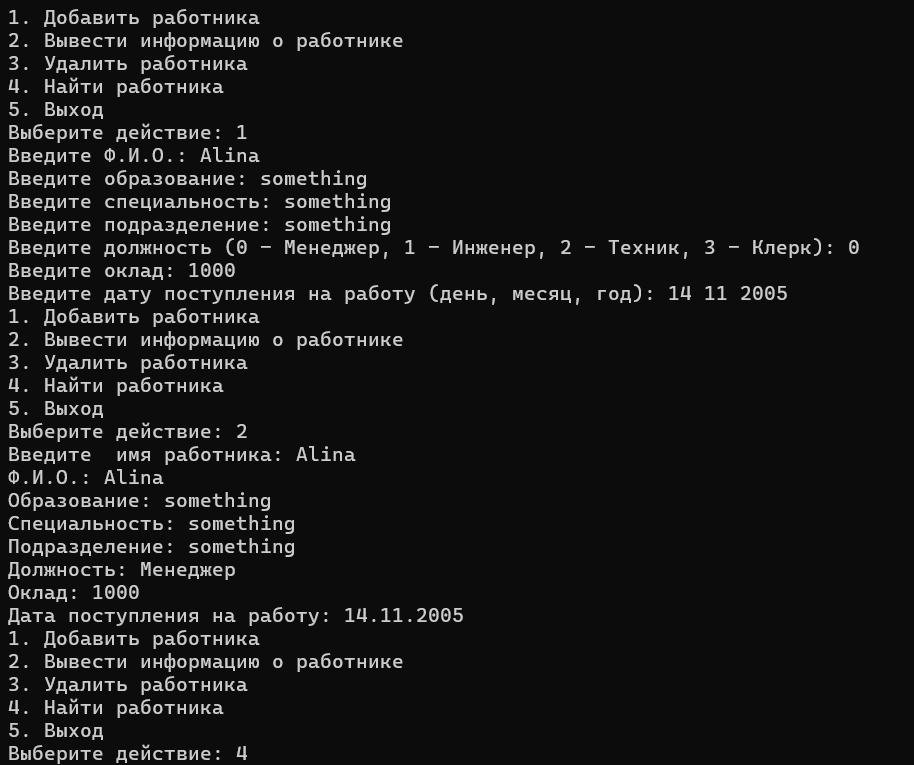
default:

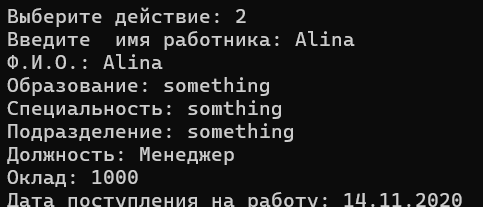
std::cout << "Неверный выбор\n";

}

}

}





Задание 2

В соответствии со своим вариантом разработать программу с использованием структуры в виде ***объединения****,* для работы с данными из таблицы, приведенной ниже. Реализовать функции ввода с клавиатуры, записи в файл, вывода на экран, чтения из файла и поиска.



#include <iostream>

#include <fstream>

#include <string>

struct Teacher {

std::string surname;

std::string examName;

std::string examDate;

};

void input(Teacher& teacher) {

std::cout << "Введите фамилию преподавателя: ";

std::cin >> teacher.surname;

std::cout << "Введите название экзамена: ";

std::cin >> teacher.examName;

std::cout << "Введите дату экзамена: ";

std::cin >> teacher.examDate;

}

void writeToFile(const Teacher& teacher, const std::string& filename) {

std::ofstream file(filename, std::ios::app);

if (file.is\_open()) {

file << teacher.surname << " " << teacher.examName << " " << teacher.examDate << "\n";

file.close();

}

}

void readFromFile(const std::string& filename) {

std::ifstream file(filename);

if (file.is\_open()) {

std::string line;

while (getline(file, line)) {

std::cout << line << "\n";

}

file.close();

}

}

void search(const std::string& surname, const std::string& filename) {

{

std::ifstream file(filename);

if (file.is\_open()) {

std::string line;

while (getline(file, line)) {

if (line.find(surname) != std::string::npos) {

std::cout << line << "\n";

}

}

file.close();

}

}

}

int main() {

setlocale(LC\_ALL, "russian");

Teacher teacher[3];

std::string filename = "teachers.txt";

for (int i = 0; i < 3; i++) {

input(teacher[i]);

}

for (int i = 0; i < 3; i++) {

writeToFile(teacher[i], filename);

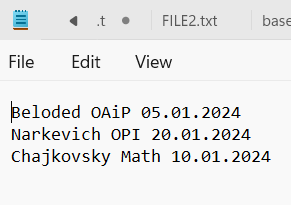
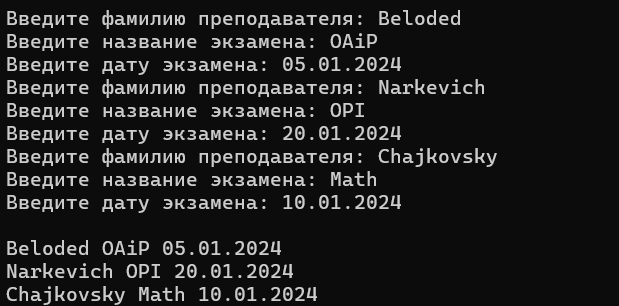
}

readFromFile(filename);

search(teacher[2].surname, filename);

return 0;

}



Дополнительные задания ( я спрашивала у Вас, и Вы разрешили сдать до 25.03.2024)

Вариант 13



#include <iostream>

#include <fstream>

#include <string>

using namespace std;

#define size 5

enum faculties {//перечисление

FIT = 1, PIM, HTIT, LHF, LID, IEF

}facults;

struct Students {//прототип структуры

string surename;

string name;

string patronimic;

string enter\_date;

string adress;

long long phone\_number;

string facult;

};

struct Students arr\_of\_structs[size];//создание массива структур

//прототипы функций

void enter\_new();

void search();

void main()//главная функция

{

setlocale(LC\_ALL, "Russian");

char choise;//переменная выбора

do {

//вывод вариантов

cout << "1-Ввести новую информацию" << endl;

cout << "2-Поиск" << endl;

cout << "3-Очистить консоль" << endl;

cout << "4-Выход" << endl;

cin >> choise;

//проверка правильности ввода

while (!(choise >= '1' && choise <= '4')) {

cout << "Некорректный ввод! Попробуйте еще раз: ";

cin >> choise;

}

int choise\_sw = choise - '0';//перевод char в int

switch (choise\_sw) {//варианты

case 1: enter\_new(); break;

case 2: search(); break;

case 3: system("cls"); break;

}

} while (choise != '4');

}

void enter\_new()//функция для ввода новой информации

{

int curr\_size = 0;//переменная для проверки количества структур

while (curr\_size < size) {//цикл для ввода новой информации

cout << "Студент номер " << curr\_size + 1 << " : " << endl;

cout << "Введите фамилию: ";

cin.ignore();//очистка буфера

//ввод информации

getline(cin, arr\_of\_structs[curr\_size].surename);

cout << "Введите имя: ";

cin.ignore();

getline(cin, arr\_of\_structs[curr\_size].name);

cout << "Введите отчество: ";

cin.ignore();

getline(cin, arr\_of\_structs[curr\_size].patronimic);

cout << "Введите дату поступления (в формате ДД.ММ.ГГГГ): ";

cin.ignore();

getline(cin, arr\_of\_structs[curr\_size].enter\_date);

cout << "Введите адрес: ";

cin.ignore();

getline(cin, arr\_of\_structs[curr\_size].adress);

cout << "Введите номер телефона: ";

cin >> arr\_of\_structs[curr\_size].phone\_number;

cout << "Введите факультет:" << endl;

//предоставление списка факультетов

int chosen\_facult;

cout << "1-FIT" << endl;

cout << "2-PIM" << endl;

cout << "3-HTIT" << endl;

cout << "4-LHF" << endl;

cout << "5-LID" << endl;

cout << "6-IEF" << endl;

cin >> chosen\_facult;

//использование switch для выбора факультета с помощью enum

switch (chosen\_facult) {

case 1: arr\_of\_structs[curr\_size].facult = "FIT"; break;

case 2: arr\_of\_structs[curr\_size].facult = "PIM"; break;

case 3: arr\_of\_structs[curr\_size].facult = "HTIT"; break;

case 4: arr\_of\_structs[curr\_size].facult = "LHF"; break;

case 5: arr\_of\_structs[curr\_size].facult = "LID"; break;

case 6: arr\_of\_structs[curr\_size].facult = "IEF"; break;

default:cout << "Неверный ввод" << endl;

}

curr\_size++;//переход к следующей карточке

system("cls");//очистка консоли

}

cout << "Вся информация успешно введена!" << endl;

cout << "Что дальше?" << endl;

}

void search()//функция поиска

{

string search\_string;//получение строки для поиска по фамилии

cout << "Введите фамилию для поиска: ";

cin.ignore();//очистка буфера

getline(cin, search\_string);//получение строки

//цикл для просмотра всего массива структур

bool check\_var = false;//создание проверочной переменной

for (int i = 0; i < size; i++) {

if (arr\_of\_structs[i].surename == search\_string) {//если условие выполнено

check\_var = true;

//вывод информации

cout << "такой студент есть!" << endl;

cout << "Фамилия: " << arr\_of\_structs[i].surename << endl;

cout << "Имя: " << arr\_of\_structs[i].name << endl;

cout << "Отчество: " << arr\_of\_structs[i].patronimic << endl;

cout << "Дата поступления в университет: " << arr\_of\_structs[i].enter\_date << endl;

cout << "Адрес: " << arr\_of\_structs[i].adress << endl;

cout << "Номер телефона: " << arr\_of\_structs[i].phone\_number << endl;

cout << "Факультет: " << arr\_of\_structs[i].facult << endl;

}

}

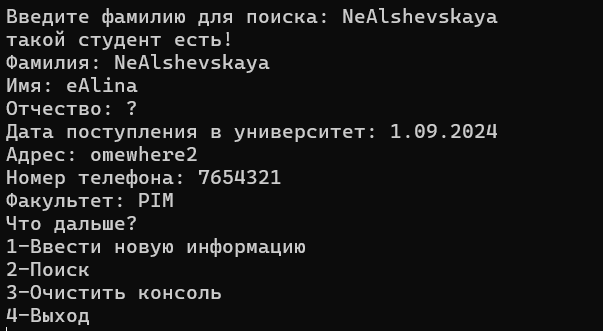
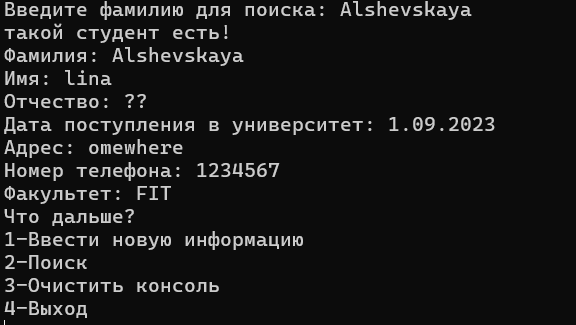
if (check\_var == false) {//если условие не выполнено

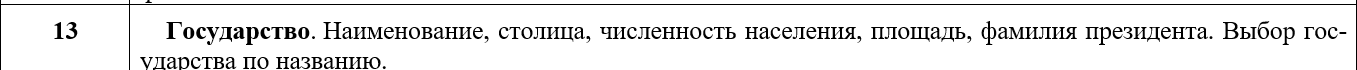
cout << "Такого студента нет." << endl;

}

cout << "Что дальше?" << endl;

}





#include <iostream>//библиотеки

#include <string>

#include <fstream>

#define size 1

using namespace std;

union state {//объединение

char name[50];

char capital[50];

long long people;

double square;

char president[50];

}state;

//прототип функции

void enter\_new();

void search();

void main()//главная функция

{

setlocale(LC\_ALL, "Russian");

char choise;

do {

//вывод вариантов

cout << "1-Ввести новую информацию" << endl;

cout << "2-Поиск" << endl;

cout << "3-Очистить консоль" << endl;

cout << "4-Выход" << endl;

cin >> choise;

//проверка ввода

while (!(choise >= '1' && choise <= '4')) {

cout << "Неверный ввод! Попробуйте еще раз: ";

cin >> choise;

}

int choise\_sw = choise - '0';//преобразование char в int

switch (choise\_sw) {

case 1: enter\_new(); break;

case 2: search(); break;

case 3: system("cls"); break;

}

} while (choise != '4');

}

void enter\_new()//функция для ввода новой информации

{

ofstream to\_file("file.txt");//открыть файл

int curr\_size = 0;

//цикл для ввода новой информации

while (curr\_size < size) {

//ввод новой информации

cout << "Для государства номер " << curr\_size + 1 << " : " << endl;//подсчет номера государства

cout << "Введите название государства: ";

cin.ignore(cin.rdbuf()->in\_avail());

cin.getline(state.name, 50);

to\_file << state.name << ';';

cout << "Введите столицу государства: ";

cin.ignore(cin.rdbuf()->in\_avail());

cin.getline(state.capital, 50);

to\_file << state.capital << ';';

cout << "Введите население: ";

cin >> state.people;

to\_file << state.people << ';';

cout << "Введите площадь государства: ";

cin >> state.square;

to\_file << state.square << ';';

cout << "Введите фамилию президента: ";

cin.ignore(cin.rdbuf()->in\_avail());

cin.getline(state.president, 50);

to\_file << state.president << ';' << endl;//переход к следующей строке в файле

curr\_size++;

}

cout << "Вся информация успешно введена!" << endl;

cout << "Что дальше?" << endl;

to\_file.close();//закрыть файл

}

void search()//функция поиска

{

ifstream from\_file("file.txt");//открыть файл для чтения

string search\_string;

cout << "Введите название государства для поиска: ";

cin.ignore();//очистить буфер

getline(cin, search\_string);//получение строки

string curr\_string;

//цикл для проверки всех строк файла

bool check\_var = false;

while (getline(from\_file, curr\_string)) {

string check\_name;

int curr\_digit = 0;

//цикл для получения только имени

while (curr\_string[curr\_digit] != ';') {

check\_name += curr\_string[curr\_digit];

curr\_digit++;

}

if (check\_name == search\_string) {//если такое государство найдено

check\_var = true;

int amm\_of\_words = 0;

int curr\_length = curr\_string.length();//получение длины текущей строки

for (int i = 0; i < curr\_length; i++) {

if (curr\_string[i] == ';') {//подсчет количества слов

amm\_of\_words++;

}

}

string\* arr\_of\_words = new string[amm\_of\_words];//создание массива слов

int curr\_word = 0;

//заполнение массива только словами

for (int i = 0; i < curr\_length; i++) {

if (curr\_string[i] != ';') {

arr\_of\_words[curr\_word] += curr\_string[i];

}

else {

curr\_word++;

}

}

//вывод требуемой информации

cout << "Такое государство есть!" << endl;

cout << "Название" << arr\_of\_words[0] << endl;

cout << "Столица: " << arr\_of\_words[1] << endl;

cout << "Население: " << arr\_of\_words[2] << endl;

cout << "Площадь: " << arr\_of\_words[3] << endl;

cout << "Фамилия президента: " << arr\_of\_words[4] << endl;

}

}

if (check\_var = false) {//если требуемое государство не найдено

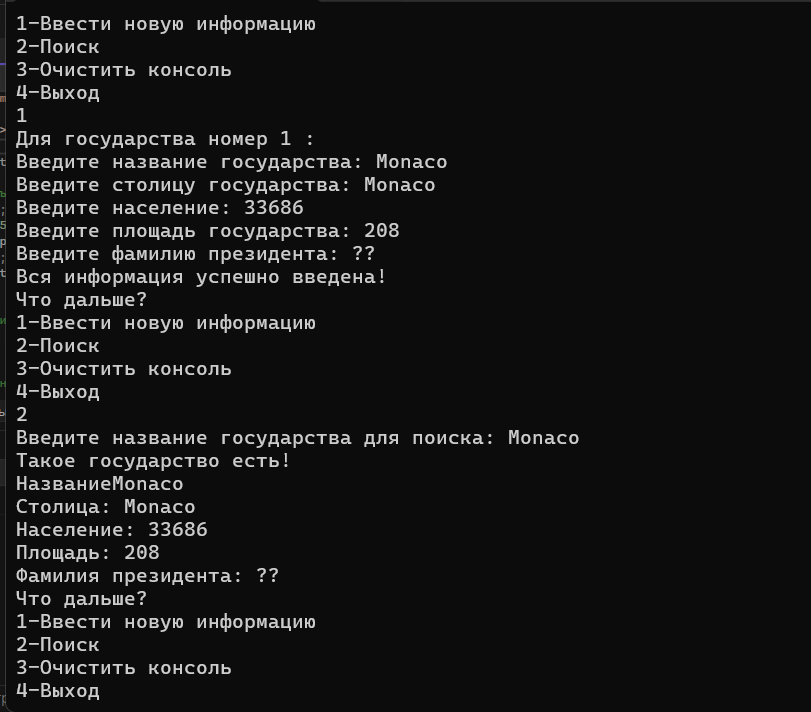
cout << "Такого государства нет." << endl;

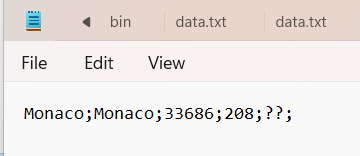
}

cout << "Что дальше?" << endl;

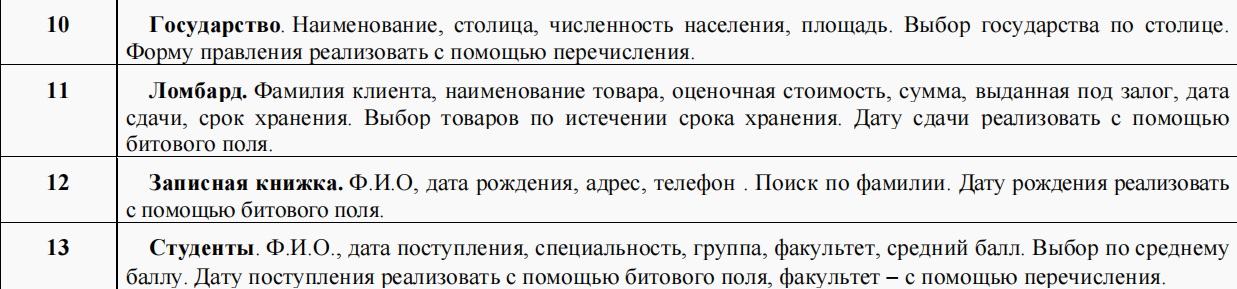
from\_file.close();//закрыть файл

}





Вариант 10



#include <iostream>

#include <Windows.h>

#include <cstring>

using namespace std;

#define LIMIT 100 // максимальное количество стран для массива структур

// перечисление

enum FormOfGovernment {

MONARCHY = 0,

REPUBLIC

};

// структура

struct Government {

char name[16];

char capital[16];

unsigned int population;

unsigned int area;

char president[16];

FormOfGovernment government;

};

Government country[LIMIT]; // массив country из объектов country

int current\_size = 0; // текущий размер массива

// прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: // Ввод элементов структуры с клавиатуры

inputInfo();

break;

case 2: // Вывод элементов структуры в консольное окно

outputInfo();

break;

case 3: // Удаление заданной структурированной переменной

deleteInfo();

break;

case 4: // Поиск информации

searchInfo();

break;

case 0: // Выход из программы

exit(0);

default:

cout << "Введено некорректное значение. " << endl;

break;

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Введите название: ";

cin >> country[current\_size].name;

cout << "Введите столицу: ";

cin >> country[current\_size].capital;

cout << "Введите численность населения: ";

cin >> country[current\_size].population;

cout << "Введите площадь территории (в км^2): ";

cin >> country[current\_size].area;

cin.ignore();

cout << "Введите фамилию правителя страны: ";

cin.getline(country[current\_size].president, 50);

int government;

cout << "Выберите форму правления страны (0 - монархия, 1 - республика): ";

cin >> government;

country[current\_size].government = static\_cast<FormOfGovernment>(government);

cout << "Элемент структуры создан. " << endl;

current\_size++; // увеличиваем current\_size после заполнения всех полей нового элемента массива

}

else { // иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

for (int i = 0; i < current\_size; i++) { // поочередно выводим информацию о странах

cout << "\nСтрана №" << i + 1 << endl;

cout << "Название: " << country[i].name << endl;

cout << "Столица: " << country[i].capital << endl;

cout << "Численность населения: " << country[i].population << endl;

cout << "Площадь страны: " << country[i].area << endl;

cout << "Правитель страны: " << country[i].president << endl;

cout << "Форма правления страны: " << (country[i].government == MONARCHY ? "монархия" : "республика") << endl;

}

}

void searchInfo() {

char capital\_name[20]; // переменная для поиска по столице

cout << "По какой столице вы хотите найти информацию? ";

cin >> capital\_name;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (strcmp(capital\_name, country[i].capital) == 0) { // если введенная страна найдена, то выводим информацию о ней

cout << "Информация о стране по выбранной столице: " << capital\_name << endl;

cout << "Столица: " << country[i].capital << endl;

cout << "Численность населения: " << country[i].population << endl;

cout << "Площадь страны: " << country[i].area << endl;

cout << "Правитель страны: " << country[i].president << endl;

cout << "Форма правления: " << (country[i].government == MONARCHY ? "монархия" : "республика") << endl;

found = true;

break;

}

}

if (!found) {

cout << "Страна не найдена." << endl;

}

}

void deleteInfo() {

char capital\_name[20]; // переменная для поиска по столице

cout << "Введите столицу страны, которую хотите удалить из структуры: ";

cin >> capital\_name;

bool found = false;

for (int i = 0; i < current\_size; i++) {

if (strcmp(capital\_name, country[i].capital) == 0) { // если введенная страна найдена, то удаляем ее из структуры

found = true;

for (int j = i; j < current\_size - 1; j++) {

country[j] = country[j + 1];

}

current\_size--;

cout << "Структурная переменная удалена." << endl;

break;

}

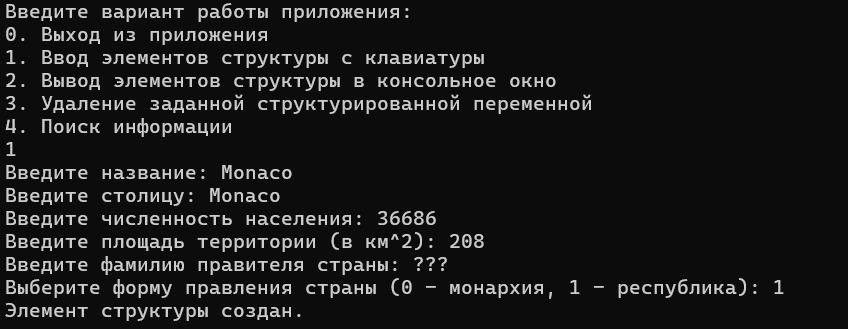
}

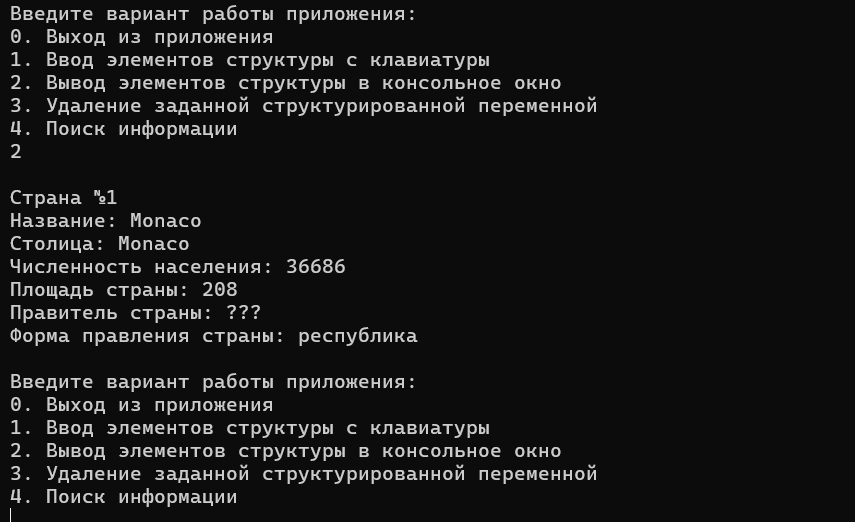
if (!found) {

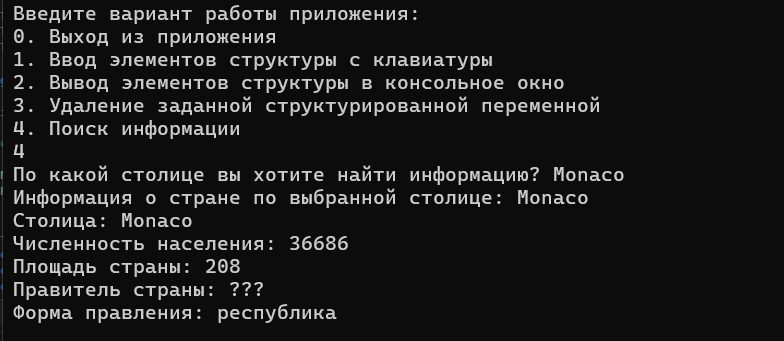
cout << "Страна не найдена." << endl;

}

}







Задание 2



#include <iostream>

#include <Windows.h>

#include <fstream>

#include <string>

#define LIMIT 100 // максимальное количество элементов для массива структур

using namespace std;

// объединение

union Notebook {

char full\_name[50];

char birth\_date[16];

char address[40];

char gender[2];

};

//структура

struct Records {

string full\_name;

string birth\_date;

string address;

string gender;

} records[LIMIT];

Notebook note[LIMIT]; // массив структур

int current\_size = 0; // текущий размер массива

// Прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации по году рождения\n";

cin >> choice;

switch (choice) {

case 1: { // Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { // Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { // Поиск информации по году рождения

searchInfo();

break;

}

case 0: { // Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // Если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько записей хотите сделать? - ";

short int amount;

cin >> amount;

cin.ignore(); // Очистка буфера ввода

ofstream ofile("notes.txt"); // Открыть файл для добавления новых данных

if (!ofile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

// Ввод информации с клавиатуры и запись ее в файл

for (int i = 0; i < amount; i++) {

cout << "Запись №" << i + 1 << ": " << endl;

cout << "Введите ФИО: ";

cin.getline(note[i].full\_name, 50);

records[i].full\_name = note[i].full\_name;

ofile << note[i].full\_name << ";";

cout << "Введите дату рождения (формат ДД.ММ.ГГГГ): ";

cin.getline(note[i].birth\_date, 16);

records[i].birth\_date = note[i].birth\_date;

ofile << note[i].birth\_date << ";";

cout << "Введите адрес: ";

cin.getline(note[i].address, 40);

records[i].address = note[i].address;

ofile << note[i].address << ";";

cout << "Введите пол (м/ж): ";

cin >> note[i].gender;

records[i].gender = note[i].gender;

cin.ignore(); // Очистка буфера ввода

ofile << note[i].gender << endl;

current\_size++;

}

cout << "Информация успешно записана в файл" << endl;

ofile.close(); // Закрытие файла

}

else { // Иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile.getline(note[i].full\_name, 50, ';')) {

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << note[i].full\_name << endl;

ifile.getline(note[i].birth\_date, 16, ';');

cout << "Дата рождения: " << note[i].birth\_date << endl;

ifile.getline(note[i].address, 40, ';');

cout << "Адрес: " << note[i].address << endl;

ifile >> note[i].gender;

cout << "Пол: " << note[i].gender << endl;

i++;

}

ifile.close();

}

void searchInfo() {

string year; // Переменная для поиска по году рождения

cout << "Введите год рождения для поиска: ";

cin >> year;

bool found = false; //по умолчанию считается, что год не найден

for (int i = 0; i < current\_size; i++) {

if (records[i].birth\_date.find(year) != string::npos) { // Если введенный год рождения найден, то выводим информацию о нем

cout << "\nИнформация по году рождения " << year << ":" << endl;

cout << "ФИО: " << records[i].full\_name << endl;

cout << "Дата рождения: " << records[i].birth\_date << endl;

cout << "Адрес: " << records[i].address << endl;

cout << "Пол: " << records[i].gender << endl;

found = true; // устанавливаем, что год рождения найден

}

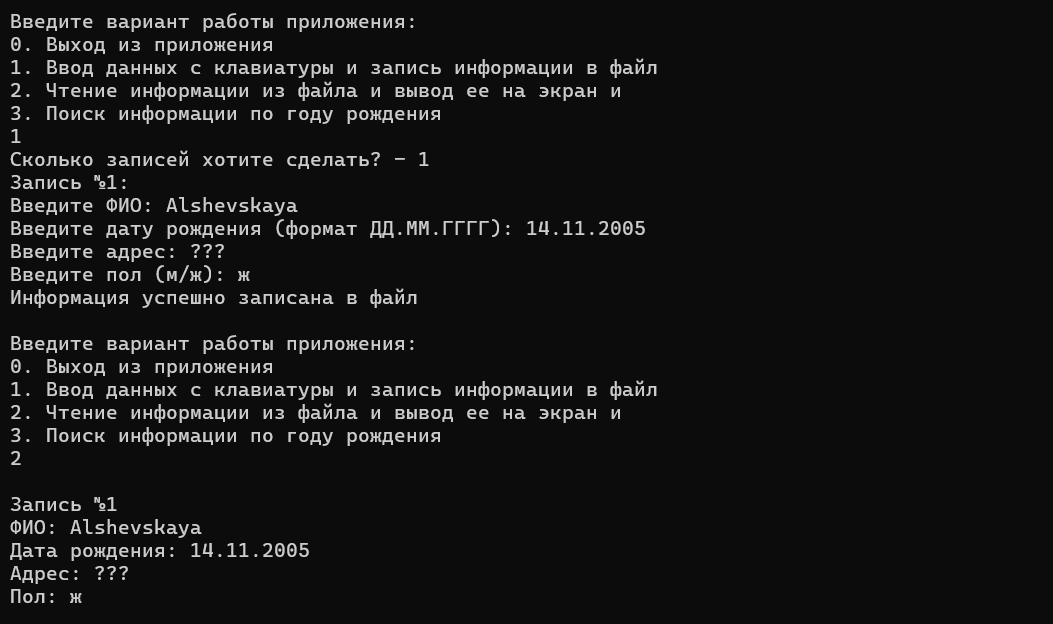
}

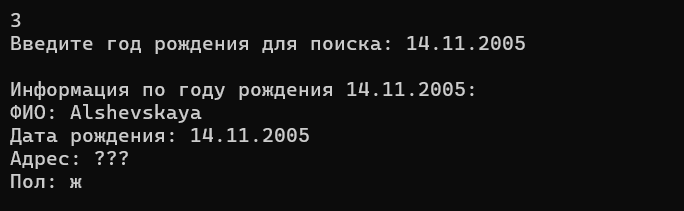
if (!found) { // Иначе выводим сообщение об ошибке

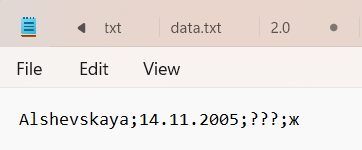
cout << "Нет информации о годе рождения " << year << endl;

}

}







Вариант 12

#include <iostream>

#include <Windows.h>

#include <cstring>

#define LIMIT 100 // ограничение по размеру массива структур

//прототипы функций

using namespace std;

void inputInfo();

void outputInfo();

void searchInfo();

void deleteInfo();

// структура

struct Notebook

{

char full\_name[80];

struct birth\_date {

unsigned day : 5;

unsigned month : 4;

unsigned year : 12;

} birthday;

char address[40];

char phone\_number[15];

};

//массив contact структуры Notebook

Notebook contact[LIMIT];

int current\_size = 0; // текущий размер массива

// главная фукнция

int main()

{

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод элементов структуры с клавиатуры\n";

cout << "2. Вывод элементов структуры в консольное окно\n";

cout << "3. Удаление заданной структурированной переменной\n";

cout << "4. Поиск информации\n";

cin >> choice;

switch (choice)

{

case 1: { //Ввод элементов структуры с клавиатуры

inputInfo();

break;

}

case 2: { //Вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { //Удаление заданной структурированной переменной

deleteInfo();

break;

}

case 4: { //Поиск информации

searchInfo();

break;

}

case 0: { //Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

}

// ввод информации

void inputInfo()

{

if (current\_size < LIMIT) {

// фио

cout << "Введите ФИО: " << endl;

cin.ignore(); // очищаем буфер

cin.getline(contact[current\_size].full\_name, 50);

// др

int day, month, year;

cout << "Введите дату рождения в формате ДД ММ ГГГГ: " << endl;

cin >> day >> month >> year; //сначала записываем значения в переменные

contact[current\_size].birthday.day = day; // а потом добавляем их в битовые поля

contact[current\_size].birthday.month = month;

contact[current\_size].birthday.year = year;

cin.ignore(); // очищаем буфер после ввода даты рождения

// адрес

cout << "Введите адрес: " << endl;

cin.getline(contact[current\_size].address, 100);

// номер телефона

cout << "Введите номер телефона: " << endl;

cin >> contact[current\_size].phone\_number;

cin.ignore(); // очищаем буфер после ввода номера телефона

current\_size++;

cout << "Элемент структуры создан. " << endl;

}

else { //иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

// вывод информации

void outputInfo()

{

for (int i = 0; i < current\_size; i++) { //поочередно выводим информацию о странах

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << contact[i].full\_name << endl;

cout << "Дата рождения: " << contact[i].birthday.day << "." << contact[i].birthday.month << "." << contact[i].birthday.year << endl;

cout << "Адрес: " << contact[i].address << endl;

cout << "Номер телефона: " << contact[i].phone\_number << endl;

}

}

// поиск информации

void searchInfo() {

char last\_name[20]; //переменная для поиска фамилии

cout << "Введите фамилию: " << endl;

cin >> last\_name;

bool flag = false; // по умолчанию считается, что фамилия не найдена

for (int i = 0; i < current\_size; i++) {

if (strstr(contact[i].full\_name, last\_name) != nullptr) { //если введенная фамилия найдена, то выводим информацию по ней

flag = true; // если фамилия найдена, то устанавливаем флажок

cout << "Информация по фамилии: " << last\_name << endl;

cout << "ФИО: " << contact[i].full\_name << endl;

cout << "Дата рождения: " << contact[i].birthday.day << "." << contact[i].birthday.month << "." << contact[i].birthday.year << endl;

cout << "Адрес: " << contact[i].address << endl;

cout << "Номер телефона: " << contact[i].phone\_number << endl;

}

}

if (!flag) { //иначе выводим сообщение об ошибке

cout << "Фамилия не была найдена" << endl;

}

}

// удаление информации по заданной переменной

void deleteInfo() {

char last\_name[20]; //переменная для поиска фамилии

cout << "Введите фамилию, которую хотите удалить из структуры: " << endl;

cin >> last\_name;

bool flag = false; // по умолчанию считается, что фамилия не найдена

for (int i = 0; i < current\_size; i++) {

if (strstr(contact[i].full\_name, last\_name) != nullptr) { //если введенная фамилия найдена, то удаляем ее из структуры

flag = true; // если фамилия найдена, то устанавливаем флажок

for (int j = i; j < current\_size - 1; j++) {

contact[j] = contact[j + 1];

}

current\_size--;

cout << "Структурная переменная очищена. " << endl;

}

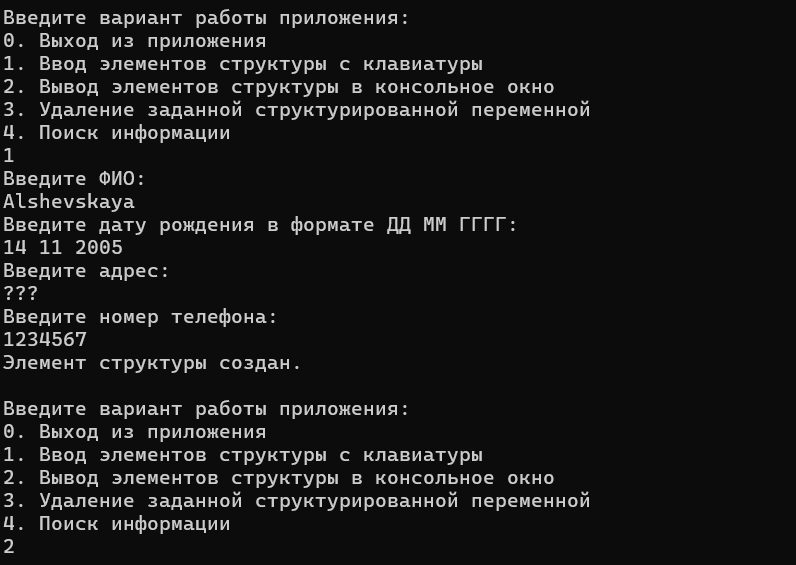
}

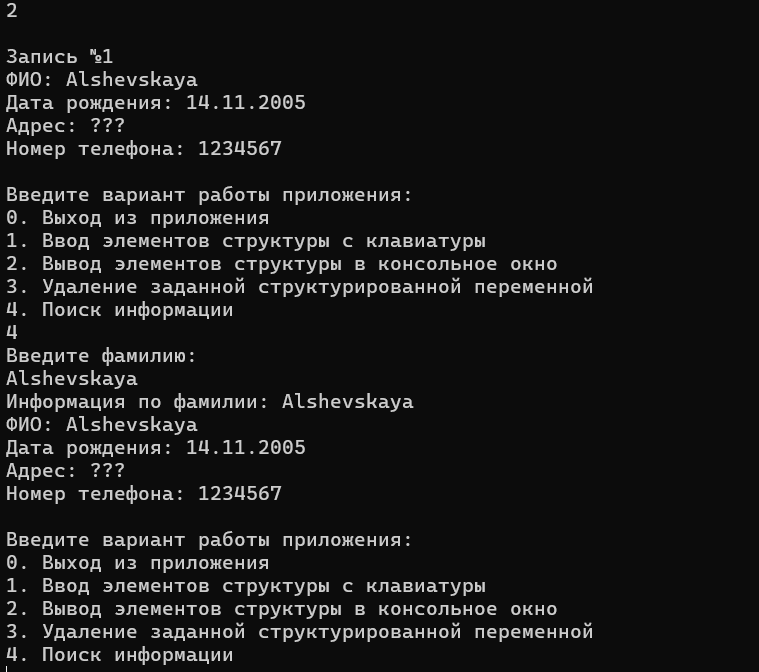
if (!flag) { //иначе выводим сообщение об ошибке

cout << "Фамилия не найдена";

}

}





Задание 2



#include <iostream>

#include <Windows.h>

#include <fstream>

#define LIMIT 100 // максимальное количество элементов для массива структур

using namespace std;

// Структура записной книжки

struct Notebook {

char full\_name[50];

char birth\_date[16];

char address[40];

char phone\_number[15];

char workplace[40];

char position[40];

};

Notebook note[LIMIT]; // массив структур

int current\_size = 0; // текущий размер массива

// Прототипы функций

void inputInfo();

void outputInfo();

void searchInfo();

int main() {

SetConsoleOutputCP(1251); // для распознавания кириллицы при записи в массив

SetConsoleCP(1251);

int choice;

do {

cout << "\nВведите вариант работы приложения: \n";

cout << "0. Выход из приложения\n";

cout << "1. Ввод данных с клавиатуры и запись информации в файл\n";

cout << "2. Чтение информации из файла и вывод ее на экран и\n";

cout << "3. Поиск информации\n";

cin >> choice;

switch (choice) {

case 1: { // Ввод элементов структуры с клавиатуры и запись в файл

inputInfo();

break;

}

case 2: { // Чтение из файла и вывод элементов структуры в консольное окно

outputInfo();

break;

}

case 3: { // Поиск информации

searchInfo();

break;

}

case 0: { // Выход из программы

exit(0);

break;

}

default: {

cout << "Введено некорректное значение. " << endl;

break;

}

}

} while (choice != 0);

return 0;

}

void inputInfo() {

if (current\_size < LIMIT) { // Если массив не выходит за пределы заданного, то вводим соотв. информацию

cout << "Сколько записей хотите сделать? - ";

short int amount;

cin >> amount;

cin.ignore(); // Очистка буфера ввода

ofstream ofile("notes.txt"); // Открыть файл для добавления новых данных

if (!ofile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

// Ввод информации с клавиатуры и запись ее в файл

for (int i = 0; i < amount; i++) {

cout << "Запись №" << i + 1 << ": " << endl;

cout << "Введите ФИО: ";

cin.getline(note[current\_size].full\_name, 50);

ofile << note[current\_size].full\_name << ";";

cout << "Введите дату рождения: ";

cin.getline(note[current\_size].birth\_date, 16);

ofile << note[current\_size].birth\_date << ";";

cout << "Введите адрес: ";

cin.getline(note[current\_size].address, 40);

ofile << note[current\_size].address << ";";

cout << "Введите телефон: ";

cin.getline(note[current\_size].phone\_number, 15);

ofile << note[current\_size].phone\_number << ";";

cout << "Введите место работы или учебы: ";

cin.getline(note[current\_size].workplace, 40);

ofile << note[current\_size].workplace << ";";

cout << "Введите должность: ";

cin.getline(note[current\_size].position, 40);

ofile << note[current\_size].position << ";";

ofile << "\n";

current\_size++;

}

cout << "Информация успешно записана в файл" << endl;

ofile.close(); // Закрытие файла

}

else { // Иначе выводим сообщение о переполнении

cout << "Структура переполнена!" << endl;

}

}

void outputInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

int i = 0;

while (ifile.getline(note[i].full\_name, 50, ';')) {

cout << "\nЗапись №" << i + 1 << endl;

cout << "ФИО: " << note[i].full\_name << endl;

ifile.getline(note[i].birth\_date, 16, ';');

cout << "Дата рождения: " << note[i].birth\_date << endl;

ifile.getline(note[i].address, 40, ';');

cout << "Адрес: " << note[i].address << endl;

ifile.getline(note[i].phone\_number, 15, ';');

cout << "Телефон: " << note[i].phone\_number << endl;

ifile.getline(note[i].workplace, 40, ';');

cout << "Место работы или учебы: " << note[i].workplace << endl;

ifile.getline(note[i].position, 40, ';');

cout << "Должность: " << note[i].position << endl;

i++;

}

ifile.close();

}

void searchInfo() {

ifstream ifile("notes.txt"); // Создание объекта класса ifstream для чтения

if (!ifile) { // В случае ошибки открытия файла

cout << "Файл не удалось открыть!" << endl;

exit(1);

}

char last\_name[20]; // Переменная для поиска по фамилии

cout << "По какой фамилии вы хотите найти информацию? " << endl;

cin >> last\_name;

bool found = false; // По умолчанию считается, что фамилия не найдена

while (ifile.getline(note[current\_size].full\_name, 50, ';')) {

if (strstr(note[current\_size].full\_name, last\_name) != nullptr) { // Если введенная фамилия найдена, то выводим информацию о ней

cout << "Информация по введенной фамилии: " << last\_name << endl;

ifile.getline(note[current\_size].birth\_date, 16, ';');

cout << "Дата рождения : " << note[current\_size].birth\_date << endl;

ifile.getline(note[current\_size].address, 40, ';');

cout << "Адрес: " << note[current\_size].address << endl;

ifile.getline(note[current\_size].phone\_number, 15, ';');

cout << "Телефон: " << note[current\_size].phone\_number << endl;

ifile.getline(note[current\_size].workplace, 40, ';');

cout << "Место учебы или должность: " << note[current\_size].workplace << endl;

ifile.getline(note[current\_size].position, 40, ';');

cout << "Должность: " << note[current\_size].position << endl;

found = true; // Устанавливаем, что фамилия найдена

break;

}

}

if (!found) { // Иначе выводим сообщение об ошибке

cout << "Фамилия была введена неверно" << endl;

}

ifile.close();

}

